## <u>CHANGES IMPLEMENTED SINCE DESIGN PHASE</u>

My database ended up representing a hypothetical repository of information related to careers rather than a CMS. My final project design ended up reflecting this more so, since I did not really understand how a CMS would be reflected in a database, so I decided to stick with what I had designed.

There were some small changes made to the database design to facilitate implementation. ‘Universities’ and ‘Bootcamp’ entities were deemed unnecessary and were causing other entities to become overloaded with foreign keys. As a result, they were condensed into one ‘institutions’ entity. I also removed some attributes related to location, since they would have required further normalisation and location data was not directly relevant to my user tasks that my database aimed to support.

Finally, some case sensitivity issues were addressed and I decided to put everything in my database in lower case. This is to allow for reduction in errors for future querying.

## <u>MY QUERIES</u>

```
-- QUERY 1: TO SUPPORT THE TASK OF CREATING A USER PROFILE, I JUST USE A STRAIGHTFORWARD INSERT
INSERT INTO user_items (username, email, password_, first_name, last_name)
VALUES ('rfitzs30', 'rfitzs30@pratt.edu', 'secret_password_here', 'Riain', 'Fitzsimons');

-- QUERY 2: JOINING TWO TABLES TO GET INFORMATION ON THE INDUSTRIES THAT ARTICLES ARE RELATED TO
SELECT a.article_id, a.title AS title_of_article, i.title AS relevant_industry
FROM article_Industry ai
JOIN Industry i ON ai.industry_id = i.industry_id
JOIN article a ON ai.article_id = a.article_id;

-- QUERY 3: This is a transaction to support my User Task of RSVP to event and having the database be updated with respect to logical effect on remaining tickets available.
START TRANSACTION;
SELECT number_tickets FROM event_items WHERE event_id = 2 FOR UPDATE;
SAVEPOINT ticket;
UPDATE event_items SET number_tickets = number_tickets - 1 WHERE event_id = 2;
INSERT INTO attendances (user_id, event_id) VALUES (19, 5);
COMMIT;
-- next to verify that the transaction had the effect of bringing ticket count down by one ticket.
SELECT number_tickets
FROM event_items
WHERE event_id = 2;
```

```sql
-- QUERY 4: My design stated that user permissions could be assigned. Here I attempt to use GRANT to give wide range of abilities to administrators
-- I also try to give standard users the ability to insert to the articles table, to support upload of articles.
-- My user types are represented in the database itself, as opposed to being users in the DBMS, this is somewhat hypothetical.
CREATE USER 'administrator';
CREATE USER 'standard_user';
SELECT user FROM mysql.user;
GRANT INSERT ON final_project_implementation.* TO 'administrator';
GRANT INSERT ON final_project_implementation.article TO 'standard_user';

-- QUERY 5: Next I demonstrate that a user permission can also be changed by updating their user type.
UPDATE user_roles
SET role_id = 2
WHERE user_id = 18;

-- QUERY 6: Granting permission to institutions and networkin groups to insert on event_items
CREATE USER 'networking groups';
CREATE USER 'institutions';
GRANT INSERT ON final_project_implementation.event_items TO 'networking groups';
GRANT INSERT ON final_project_implementation.event_items TO 'institutions';

-- QUERY 7: When networking networking groups and institutions create events it is facilitated by an insert.
INSERT INTO event_items (title, date_of, number_tickets, cost)
VALUES
   ('Summer Festival', '2024-07-15', 500, 25.00),
   ('Tech Summit', '2024-09-20', 300, 50.00),
   ('Art Exhibition', '2024-10-10', 200, 10.00);

-- QUERY 8: My user tasks say people can 'browse' events. I will create a view that reflects this kind of information retrieval for a user looking for most affordable events.
CREATE VIEW least_expensive_events_view AS
SELECT
   event_id,
   title AS event_title,
   date_of AS event_date,
   number_tickets AS event_tickets_available,
   cost AS event_cost
FROM
   event_items
ORDER BY
   cost
LIMIT 4;
SELECT * FROM least_expensive_events_view;
```

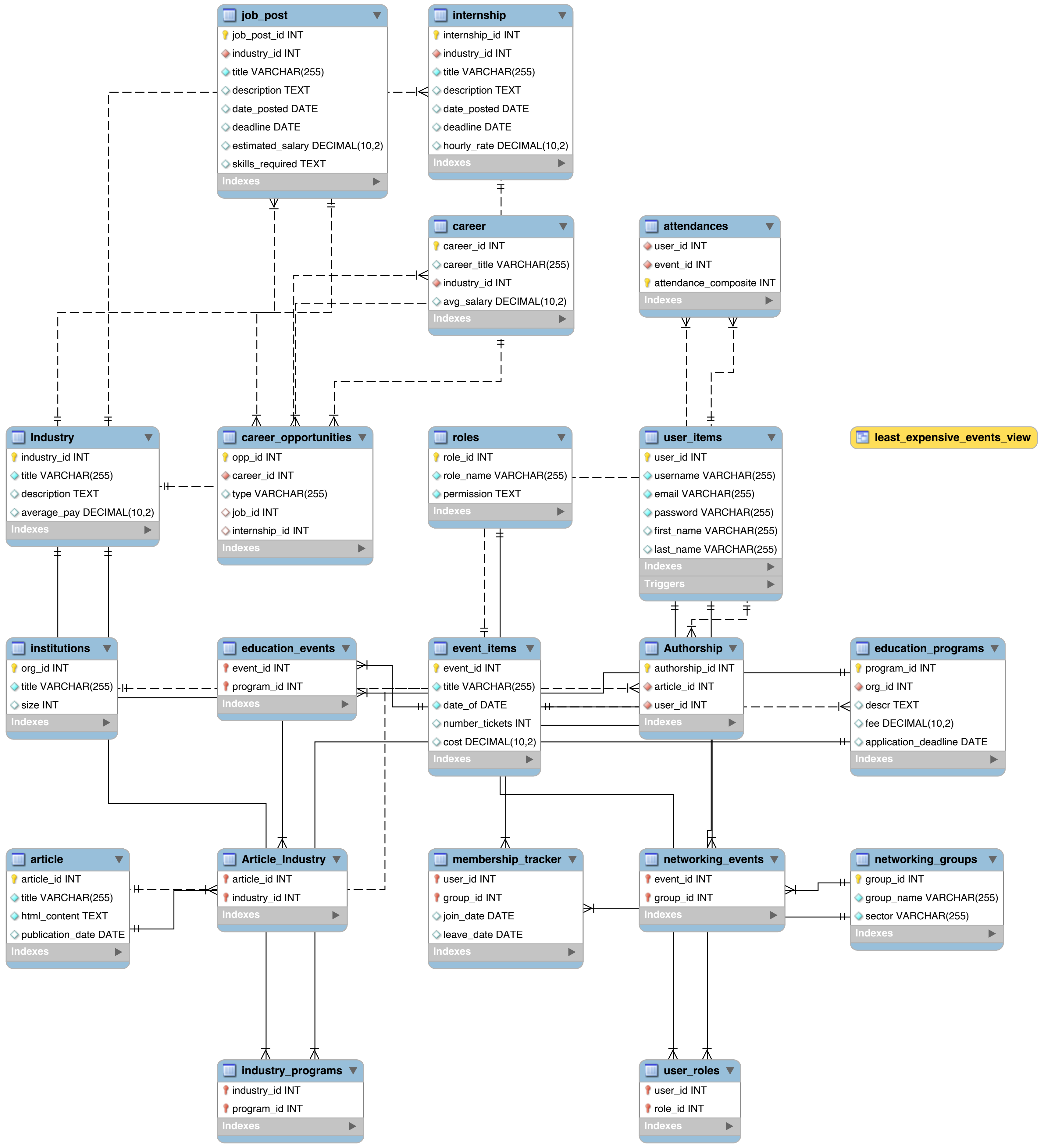
```
-- QUERY 9: Tracking attendances is a task supported by the database. Here I use a query
with subquery to find out how well events of 2024 were attended. This
SELECT
   event_id,
   title AS event_title,
   (
      SELECT COUNT(*)
      FROM attendances
      WHERE attendances.event_id = event_items.event_id
   ) AS total_attendees
FROM event_items
WHERE YEAR(date_of) = 2024;

-- QUERY 10: Should users be deleted from the database, a trigger can make sure their
data is removed from associated tables.

CREATE TRIGGER DeleteUser AFTER DELETE ON user_items FOR EACH ROW
BEGIN
   DELETE FROM user_roles WHERE user_id = OLD.user_id;
   DELETE FROM attendances WHERE user_id = OLD.user_id;
   DELETE FROM Authorship WHERE user_id = OLD.user_id;
END;

-- QUERY 11: Users can leave groups
DELETE FROM membership_tracker
WHERE user_id = 10
AND group_id = 10;
```

**PHYSICAL DESIGN (SEE NEXT PAGE)**


job_post
job_post_id INT
industry_id INT
title VARCHAR(255)
description TEXT
date_posted DATE
deadline DATE
estimated_salary DECIMAL(10,2)
skills_required TEXT
Indexes
internship
internship_id INT
industry_id INT
title VARCHAR(255)
description TEXT
date_posted DATE
deadline DATE
hourly_rate DECIMAL(10,2)
Indexes
career
career_id INT
career_title VARCHAR(255)
industry_id INT
avg_salary DECIMAL(10,2)
Indexes
attendances
user_id INT
event_id INT
attendance_composite INT
Indexes
Industry
industry_id INT
title VARCHAR(255)
description TEXT
average_pay DECIMAL(10,2)
Indexes
career_opportunities
opp_id INT
career_id INT
type VARCHAR(255)
job_id INT
internship_id INT
Indexes
roles
role_id INT
role_name VARCHAR(255)
permission TEXT
Indexes
user_items
user_id INT
username VARCHAR(255)
email VARCHAR(255)
password VARCHAR(255)
first_name VARCHAR(255)
last_name VARCHAR(255)
Indexes
Triggers
least_expensive_events_view
institutions
org_id INT
title VARCHAR(255)
size INT
Indexes
education_events
event_id INT
program_id INT
Indexes
event_items
event_id INT
title VARCHAR(255)
date_of DATE
number_tickets INT
cost DECIMAL(10,2)
Indexes
Authorship
authorship_id INT
article_id INT
user_id INT
Indexes
education_programs
program_id INT
org_id INT
descr TEXT
fee DECIMAL(10,2)
application_deadline DATE
Indexes
article
article_id INT
title VARCHAR(255)
html_content TEXT
publication_date DATE
Indexes
Article_Industry
article_id INT
industry_id INT
Indexes
membership_tracker
user_id INT
group_id INT
join_date DATE
leave_date DATE
Indexes
networking_events
event_id INT
group_id INT
Indexes
networking_groups
group_id INT
group_name VARCHAR(255)
sector VARCHAR(255)
Indexes
industry_programs
industry_id INT
program_id INT
Indexes
user_roles
user_id INT
role_id INT
Indexes